IN THE SPECIFICATION

Please amend the specification as follows. Paragraphs that are being amended are listed in their entirety; changes are indicated in the margin with a vertical change bar. Deletions are marked by ~~strikethrough~~; insertions are underlined.

Please amend the paragraph on page 4, lines 3-4, as follows:

Figure 3 is a block diagram that illustrates the Connector interface between the enterprise data sources and the mobile client of ~~Figure 1~~ Figure 1.

Please amend the paragraph beginning on page 7, lines 7-9, as follows:

Business Objects--programmable objects based on business concepts, combining fields and relating information from different enterprise data ~~sources.~~ sources (e.g. data sources such as Customer, Contacts, Assets, Tasks, etc.).

Please amend the paragraph beginning on page 14, lines 20-26, as follows:

For example, an "Oracle" applications connector allows a customer to make calls to Oracle support services, either through the closest data constructs the customer has to APIs (such as PL/SQL procedures) or directly to the enterprise database itself via ODBC. As with all of the ODBC connectors the ~~dynamically~~ dynamic interrogation of the RDBMS schema is automatically executed, exposing the specific physical design of the database. This gives the customer a hierarchical view of the actual interfaces into that system.

Please amend the paragraph beginning on page 15, line 12, through page 16, line 2, as follows:

In the metadata 312, the data definition from the enterprise data sources is mapped to views that are used to create the data store on the client and store the relevant business data on the mobile client from the enterprise data sources in a relational ~~database.~~ database. Access to this business data is performed ~~via a~~ via the business object layer defined and stored in metadata on the mobile client. As shown in Figure 3, the ORDER_ID from the ERP data source is mapped to a business object property called OrderID, whose relational definition is stored in metadata 318 on the mobile client 316 and utilized by one or more of the mobile applications also defined in metadata. The F_NAME data from the CRM enterprise data source is mapped to (stored into) the FirstName business object property definition stored in the mobile client database, and the L_NAME data is mapped to the LastName business object property. Similarly, the CRED_LIM data from the HR/Finance data source is mapped to the CreditLimit business object property, and the WARRANTY data from the Legacy/ODBC data source is mapped to the Warranty business object property. Thus, data from the potentially dissimilar and incompatible disparate enterprise data sources 302, 304, 306, 308, 310 are delivered to the mobile client through the Data Manager Web Services to the local data store (represented by the lines from the enterprise data sources to the application server 314) in the proper format for access using one of the business objects on the mobile client (indicated in the mobile client 316 with actual values).

Please amend the paragraph beginning on page 18, lines 14-19, as follows:

1: Get Latest: In this update type, the mobile client makes a request to get the latest information from the enterprise data sources via the Dexterra application server. The Dexterra application server ~~process~~ processes the request and retrieves the business information from the multiple data sources

using the Dexterra Connector Web Service and delivers the business information to the mobile client.

Please amend the paragraph on page 19, lines 17-19, as follows:

Depending on configuration of the application server, Conflicts are resolved in one of four ways: First Update Wins, Last Update Wins, Admin Resolution or Server-side ~~Rule~~ Rule.

Please amend the paragraph beginning on page 21, line 22, through page 22, line 5, as follows:

The Administrator 110 also supports security management features that provide a mechanism to add and change users, integrating with existing directory services and/or LDAP or Active Directory Services in a "two tiered" security model. The security management then "authorizes" that client based on the information received in the authentication process to determine who the client identified. Thus, a high level of security is provided, because the application is secure on the client (requires username/password for access) and the downloaded data is secure on the client, and the system access (through the SQL CE interface) is controlled. There is a "device disablement" process built ~~Into~~ into the system through the configuration desktop, and a user can be disallowed access to the device application and data. Additionally, there is a significant amount of history tracking, logging, and metrics built into the system for auditing purposes. Other models of security are supported, including IIS Authentication support and DB/third-party system level security/authentication, all over Secure Socket Layer (SSL).

Please amend the paragraph on page 24, lines 24-29, as follows:

Referring back to Figure 2, this block diagram shows how the relationships between columns and fields in the target application are related to information ~~In~~ in the "FormFlows" (steps in the business process represented as ~~'Forms"~~ "Forms" in

the application) and are then associated into the ForceFlow (the business process). There can be many Business Objects in one FormFlow and potentially more than one FormFlow in any business process.

Please amend the paragraph on page 33, lines 15-25, as follows:

b. Last Update Wins

Under the Last Update conflict resolution process 806, the application server will accept the last version of a client data change, as provided ~~by an~~ by a mobile client. As a result, for Last Update processing, the application server simply accepts the last data version provided by any mobile client, and overwrites any previous changes to the data that might have been made. Therefore, if the application server is configured to operate according to the Last Update technique, then the conflict detection operation (see box 618 processing description above) is not needed and is not performed. That is, the application server simply persists the data changes from the mobile client to the back end enterprise data source, overwriting the current record in the back end enterprise data source.

Please amend the paragraph beginning on page 37, lines 11-22, as follows:

For determining conflicts at the column level, the application server will recognize changes made to the same column of data records. The columns of two data records being checked for conflict will be examined column-by-column for any changes. For example, suppose a first client (Client "A") changes the address for a particular row of a data table and synchronizes those changes with the application server. Also suppose that a second mobile client (Client "B") uploads its updated telephone number for the same data record. When the application server compares the two data records using the column-level conflict determination, the application server will recognize that although a data change has occurred on the same row, the changes exist on two different columns. For ~~the~~ example, the application server

would determine that there is no conflict, and therefore the application server would accept both changes.

Please amend the paragraph on page 44, lines 8-20, as follows:

The Customer Data Definition is the schema ~~that~~ in which the Customer Business Data is going to be stored, in a (e.g., a relational database file) Customer Business Data store on the device. The schema is translated into views on the Dexterra Server, which correspond to either tables on a customers database or objects exposed through their APIs. During the Subscription process the Dexterra Server sends down to the client device the schema definition, defined by the views ~~as~~ and SQL Create Schema statements which the device then executes to create the database definition on the client device. The Business Object definitions then contain information on how to retrieve data out of the database to allow the application to operate. The schema is by default defined from the customer's backend system, and then directly delivered to the client device. In cases where the schema cannot be obtained from the backend system, system administrators have the ability to describe the schema that corresponds to the backend system.